Name: Pissima Kabadwal

TUTORIAL-2

Q.1

Time Complexity of below code

```
Void fun(int n)
{                                   → O(1)
  int j=1; i=0;
  while (i<n)
  {
    i = i+j;
    j++
  }
}
```

i=0 → i=0+1, j=2
i=1 → i=1+2, j=3
i=3 → i=1+2+3, j=4
i=6 → i=6+4, j=5
⋮
i=k → i=k+j, j=n

(k+j: Sum)

or

j	1	2	3	4	.....	n
i	1	3	6	10	.....	k

$$\frac{k(k+1)}{2} > n$$

$$\frac{k^2+k}{2} > n$$

$$k^2 = n$$

$$k = \sqrt{n}$$

T.C = O(k) or $O(\sqrt{n})$

Q.2 Write recurrence relation for recursive function that prints fibonacci series.

```
int fib (int n)
{
  if (n <= 1)
    return n;
  return fib(n-1) + fib(n-2);
}
```

$$T(n) = \begin{cases} 1 , & n <= 1 \\ T(n-1) + T(n-2) , & \text{otherwise.} \end{cases}$$

$$T(n) = T(n-1) + T(n-2) + c$$

$$= 2T(n-1) + c$$

[$T(n-1) \sim T(n-2)$ from approximation

$$T(n) = 2[2T(n-2) + c] + c$$

$$= 4T(n-2) + 3c$$

$$= 8T(n-3) + 4c$$

$$\vdots$$

$$2^k T(n-k) + 2^k - 1)c$$

$T(1) = 1$

$T(n-k) = T(1)$

$n = k+1$

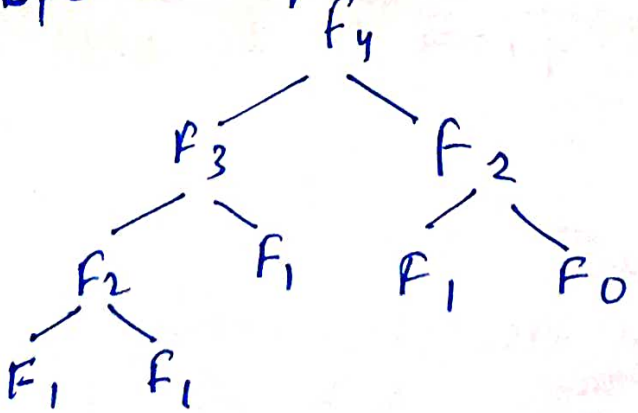
$\boxed{k = n-1}$

$$T(n) = 2^{n-1} T(n-k+1) + (2^{n-1} - 1)\, c$$

$\boxed{\text{Time complexity} = O(2^n)}$

for space complexity:

Space required $\propto$ max. dep



$\rightarrow O(4)$

$\rightarrow$ for n elements $O(n)$

Q.3 Write programs which have complexity

1) n log n

```
A()
{ int i, j;
    for(i=1; i<=n; i++) → O(n)
      {
        for(j=1; j<=n; j=j/2) → O(log n)
          {
            printf("#");
          }
      }
}
```

2) n^3

```
A()
{ int i,j,k;
  for(int i=0; k=n; i++) → O(n)
      {
        for(j=0; k<=n; j++) → O(n)
          {
            for(k=0; k<=n; k++) → O(n)
            { printf("*");
            }
          }
      }
```

Name: Purnima Kabadwal

Q.4 Solve the following recurrence relations.

$$T(n) = T(n/4) + T(n/2) + cn^2$$

$$= T(n/4) \leq T(n/2)$$

$$T(n) = T(n/2) + T(n/2) + cn^2$$

$$= 2T(n/2) + cn^2$$

Comparing with master's eq^n we get

$a = 2$
$b = 2$
$c = \log_2 2^2 = 1$
$n < n^2$

$k = 2$
$p = 0$

$a < b^k$
$O(n^k \log^p n)$
$O(n^2 (\log n)^0)$

$$\boxed{Tc = O(n^2)}$$

Question 5:-

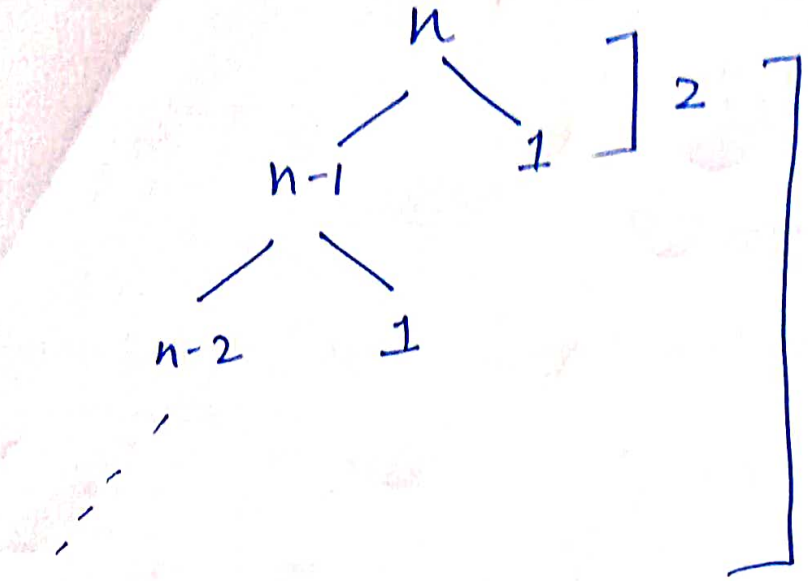
What is time complexity of following f^n.

```
function fun ()
int  func int n) {
   for (int i = 1; i <= n ; j++) {
     for (int j = 1; j < n; j += i)
     { // O (i) // task
     }
   }
}
```

Incrementation depends on i: well inner all loop.

Question 7

$$T(n) = T(n-1) + O(1)$$



$$T(n) = T(n-1) + T(n-2) + \ldots T(1) + O(1)) \times n$$

$$= n \times n$$

$$\boxed{T(n) = O(n^2)}$$

Lowest Height = 2

Highest = n

$$\boxed{\text{Diff} = n-2}, \quad n > 1$$

Given algo produces linear search.

Question 6.

Time complexity of

```
for(int i=2; i<=n; i=pow(i,k))
{
    // O(1)
}
```

$i = 2$ $\quad$ $i = 2^k$ $\quad$ $i = (2^k)^k = 2^{k^2}$

'2 to n times' $\quad$ '2^k to n times' $\quad$ 2^{k^2} to n times.

$$i = 2^{k \log_k (\log n)} = 2^{\log_2 n} = n$$

Total TC → $O(\log \log n)$

$$2^{k^l} = n$$

$$k^l = \log_2 n$$

$$k^l = \log_2 n$$

$$l = \log_k \log_2 n$$